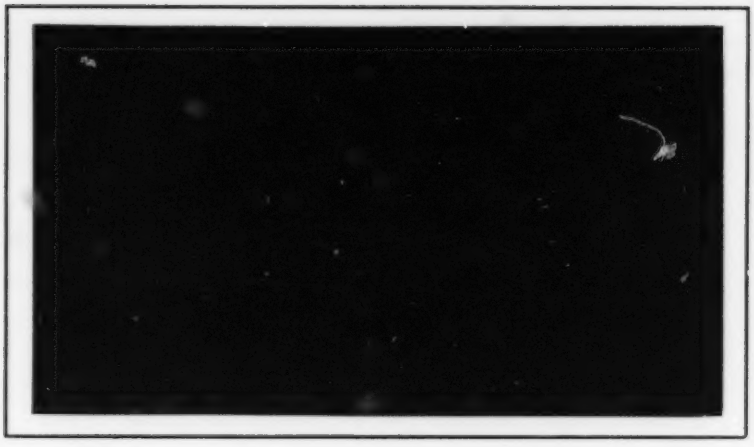

CENTRE FOR
SYSTEMS
SCIENCE

Simon Fraser University, Burnaby, B.C., Canada V5A 1S6 / (604) 291-4704

92-05398/

Research in Natural Language Processing at Simon Fraser University

Dan Fass and Nick Cercone

CSS-IS TR 91-12

November 1991

Research in Natural Language Processing at Simon Fraser University

Dan Fass and *Nick Cercone*

Centre for Systems Science
Simon Fraser University
Burnaby, British Columbia, Canada V5A 1S6.
fass@cs.sfu.ca and nick@cs.sfu.ca

Research in natural language processing (NLP) at Simon Fraser University (SFU) is described. The research is divided into applications, theory and integrated systems. The three main applications of NLP pursued at SFU are natural language interfaces, speech recognition, and machine translation. There is considerable theoretical work in syntax. Grammar theories studied intensively are government-binding theory, static discontinuity grammar, head-driven phrase structure grammar, and tree unification grammar. Integrated systems research includes integrating NLP systems with logic programming, database systems and expert systems, and designing tools for developing grammars for NLP systems.

1. Introduction

Writing a letter, reading a newspaper, watching the six o'clock news, having a conversation – the every-day written and spoken language of such activities is called *natural language* to distinguish it from artificial, made-up languages like programming languages. For over 30 years, researchers have studied how computers can be programmed to understand and generate written text and spoken utterances. The study area has been called *natural language processing* (NLP) or *computational linguistics*, though these terms tend to be associated with text processing rather than speech processing.

These days, NLP research is conducted at many universities and in the research laboratories of large companies, and there is a small but growing number of commercial NLP products (Obermeier, 1989). The commercial interest exists because the payoff for successful NLP systems is potentially enormous, particularly in applications like *natural language interfaces, machine translation systems*, and *speech recognizers*. Natural language interfaces allow people to put questions to and receive answers from machines in a natural language like English. A particular application of natural language interfaces is to computer databases so that non-technical people can directly access the information in the databases. Machine translation systems automatically translate between one natural language (say French) and others (say Spanish or English). Speech recognizers understand the meaning of a speaker's utterance or, in the case of *speech verification*, infer the identity of a speaker or check a speaker's claimed identity.

The Centre for Systems Science (CSS) at Simon Fraser University (SFU) has almost certainly the largest NLP group in Canada, and one of the larger groups in North America. SFU's NLP research can be divided into three broad categories: applications, theory, and integration with other computer systems.

- **Applications** (section 3). The three main NLP applications at SFU are natural language interfaces, speech recognition, and machine translation. The SFU group has extensive experience with these applications, especially natural language interfaces.
- **Theory** (section 4). Broadly interpreted, the theoretical NLP research at SFU is of two kinds. Some of the research is into detailed mechanisms for handling particular problems in natural language, e.g., Fred Popowich's work on reflexive pronouns. The remaining research investigates the relationships between different NLP formalisms, e.g., T. Pattabhiraman's comparison of different approaches to generating text. Both kinds of research feed into the NLP applications described in section 3: several of the approaches developed by our researchers are being used in the applications, e.g., Veronica Dahl's static discontinuity grammar.
- **Integrated Systems** (section 5). A growing research topic in NLP is the integration of NLP systems with systems that offer other facilities and capabilities. Section 5 describes SFU research into integrating NLP systems and ideas with: logic programming, tools for developing grammars for NLP systems, general-purpose interfaces, database systems, and expert systems.

The appendices contain further information about NLP research at SFU. Appendix A lists the personnel involved, Appendix B lists the hardware and software used, Appendix C describes the structure of the CSS, and Appendix D lists the grants and industrial contracts of the CSS's main NLP researchers. Some of the material in sections 3-5 and appendices A and B is adapted from material in the *CSS Update*, *CSS Research Review* and *CSS Natural Language Laboratories* leaflet, all edited by Barry Shell. The interested reader is encouraged to look at these. Carolyn Seely-Morrison, Administrative Assistant of the CSS, was of great assistance in preparing Appendix D.

2. Natural Language

This section introduces some of the major problems of natural languages faced by NLP researchers. The section can be skipped by those familiar with NLP.

The study of natural language is frequently decomposed into a number of smaller, partially overlapping study areas: phonology, morphology, syntax, semantics and pragmatics. The scope of each area is described below, together with problems that each area presents for NLP. The descriptions of areas are adapted from Crystal (1991).

• Phonology

Phonology is the study of the sound structure of language. Sounds are organized into a system of contrasts, and analyzed in terms of phonemes, distinctive features, or other such phonological units according to the theory used. A phoneme is the minimal unit of the sound system of a language. Some languages have as few as 15; others have as many as 80. No two languages have the same system of phonemes. Distinctive features are used either to define phonemes or as an alternative to the notion of phoneme. Example pairs include +nasal and -nasal, and +voice (voiced) and -voice (voiceless). Nasal sounds are produced when there is complete closure in the mouth and all the air thus escapes through the nose, as in the 'n-' sound of 'nasal'. Voiced sounds are produced while the vocal cords are vibrating, e.g., the 'b-' sound in 'bin'; voiceless or unvoiced sounds are produced when there is no such vibration, as in the 'p-' sound of 'pin'.

Problems include the ratio of noise to data, the varying speech rates within and across individuals, and coarticulation. Coarticulation takes place when the articulation for two or more sounds takes place in the vocal tract, e.g., the 'sh-' in 'shoe' is normally pronounced with lip-rounding in anticipation of the '-oo' sound.

• Morphology

Morphology is the study of the structure of words, especially through use of morphemes. Morphemes are commonly divided into free forms (morphemes which can occur as separate words) and bound forms (morphemes which cannot occur in this way, e.g., 'unselfish' consists of three morphemes, 'self' which is a free form, and 'un-' and '-ish' which are bound forms.

A major morphological problem is ambiguity: the suffix 's', for example, can indicate the plural of a noun or the present tense of a verb. Another problem is exceptions, for example, the plural of the noun 'foot' is 'feet' (not 'foots').

• Syntax

Syntax is the study of how words are combined to form sentences in a language. Syntactic structures (or constructions) are analyzed into sequences of syntactic categories (or classes). The sequences are established on the basis of syntactic relationships that linguistic items have with each other in a construction, e.g., "tall people" is generally analyzed into a noun phrase consisting of an adjective 'tall' and a noun 'people'. Parsing refers to the assignment of syntactic categories and structures in single sentences. The following are some major syntax problems.

Structural ambiguity occurs when a sentence construction can be assigned several possible structures or combinations of elements, e.g. in "Jane saw the man in the park with the telescope" the prepositional phrase "with the telescope" could be attached to either "Jane saw" or "the man in the park."

Unbounded or **long distance dependency** is a relationship between two syntactic components of a sentence in which the related constituents are not required to be within some bounded distance of each other. The dependency, which may extend over one or more clause boundaries, usually involves an empty noun phrase constituent called a "trace" which is coindexed with another noun phrase appearing earlier, as in "Show me the report$_{[i]}$ that Nick wanted Dan to write$_{[i]}$" where, although 'report' is the object of the verb 'write', there is no explicit object following the verb.

• Semantics

Semantics is the study of meaning in language. It contains a number of branches including philosophical semantics and linguistic semantics, which have both been studied in NLP. Philosophical semantics studies relations between linguistic expressions (like sentences) and the entities in the world to which they refer, and the conditions under which such expressions can be said to be true or false. Analysis is performed with logical systems. Linguistic semantics studies the semantic properties of natural languages using a variety of linguistic constructs. Among the phenomena studied within semantics are the following.

Lexical ambiguity refers to a semantic property of words that they can have multiple senses or meanings, e.g., the word 'crook' has different senses: it can mean a thief, a bend, or a shepherd's stick. Resolution of lexical ambiguity is required for understanding sentences that contain ambiguous words like 'crook', e.g., in "The crook stole a diamond ring," the thief sense is meant.

Similarity or **paraphrase** refers to a property of sentences that different ones can have the same (or very similar) meanings, e.g., "Give me the Western region outage log summary for July," "Give me the July outage log summary for the Western region," "Give me the outage log summary for July for the Western region" and "Give me the July Western region outage log summary" (McFetridge, 1991). The problem is recognizing when two sentence are paraphrases.

Reference is a relationship of identity between linguistic units, e.g., between a pronoun and a noun or noun phrase. Pronouns are of various kinds, including definite pronouns like 'it' and 'them', personal pronouns such as 'I' and 'you', reflexive pronouns like 'myself' and 'yourself', and relative pronouns such as 'who', 'whom' and 'that'. The problem is resolving reference, i.e., connecting a pronoun with the noun or noun phrase to which it refers.

Reference can occur across sentence boundaries, and can be backwards or forwards. Anaphora (or back-reference) is reference to an earlier part of a discourse. Cataphora (or forward reference) is reference to a later part of the discourse. The difference can be seen in a two different two-sentence discourses where the first sentence each time is "John is at home." There is an anaphoric reference to John when the second sentence is "If he is not drunk, Peter will be surprised" versus a cataphoric reference to Peter when the second sentence is "If he is not drunk, Peter will take me there" (Strzalkowski and Cercone, 1986, p. 159). Traditional syntactic solutions have been able to treat only simple classes of anaphora and only occasional inter-sentential references.

• Pragmatics

Pragmatics is the study of the communicative use of language, particularly the structure of conversations and dialogue: how participants take turns in conversations, how speakers use knowledge of communication (e.g., about the context in which language is used), and the effects their use of language has on other participants. Pragmatic problems include the following.

Presupposition is the information assumed by a person when using language and which is as the centre of a person's communicative interest, e.g., "There is unrest in Yugoslavia" assumes the existence of (a country called) Yugoslavia.

Conversational repair refers to "the attempt made by participants in a conversation to make good a real or imagined deficiency in the interaction (for example, a mishearing or misunderstanding)" (Crystal, 1991, p. 298). A major problem here is working out which participant is wrong or mistaken and hence should have their conversation (and understanding) repaired.

Indirect meaning refers to the communicative purpose of a piece of language which does not directly reflect its surface form. The true communicative purpose is understood from examining the context in which the piece of language was used, for example, "It's hot in here" looks like an assertion, but in the right context – spoken to someone standing by a window – might be a request to open the window. Likewise, "Can you pass the salt?" looks like a question, but can also be a request to pass the salt if said when sitting at a table and spoken to someone closer to the salt than you are!

3. Applications

• Natural language interfaces to databases

One of the most intensively studied areas of NLP is question-answering by natural language interfaces serving as "front ends" to databases. Such front ends help users in two main ways (McTear, 1987). First, they are relieved of the need to know the structure of the database. Second, the system is made more convenient and flexible for them. A particular application of natural language interfaces is executive information systems which allow management executives to access directly the information in the databases of their companies.

SystemX – Nick Cercone, Paul McFetridge, Gary Hall, Wo-Shun Luk, Chris Groeneboer.

SystemX turns ordinary English questions into database queries expressed in SQL (short for Structured Query Language), the standard computer language for manipulating relational databases (McFetridge et al, 1988a, 1988b; Cercone et al, 1989). SystemX has quite extensive coverage of English, and can handle passives, imperatives, possessives, relative clauses, prepositional phrases and quantification.

SystemX is of a modular design, containing a natural language understanding module and a database query module. The natural language understanding module has three components: a lexical analyzer (which analyzes words), parser (which does syntax processing), and semantic interpreter (which does semantic interpretation and produces canonical semantic representations).

The lexical analyzer contains two subsystems, TEMPLATE and MORPHOS, and accesses a syntactic lexicon. TEMPLATE uses the form of certain words to recognize that they belong to certain categories such as proper names, part numbers and report identity numbers. MORPHOS is a morphological analyzer which employs a set of rules to strip the endings off words and identify their roots. Words are looked up in the syntactic lexicon and the syntactic information retrieved is used by the parser.

The parser contains a set of grammar rules. The rules are applied to the syntactic information and a parse tree for the sentence is built. The parse tree is passed to the semantic interpreter.

The semantic interpreter contains a set of semantic rules and a semantic lexicon. Each entry in the lexicon is a frame containing a description of the database entity referred to by a particular word or expression. Information from the frames is attached to the parse tree and then the semantic rules are applied to build canonical semantic representations. The entities and relationships within the representations are passed to Pathfinder, a subsystem which derives the logical access path to the requested information in the database. Pathfinder uses the semantic information inherent in the relational database to assist in the disambiguation of English queries to the database. Queries to a database can be interpreted with respect to a conceptual model which represents the entities and relationships of the domain, and which can be easily mapped to the database itself. Queries (or parts of them) correspond to substructures of the conceptual model. If a query is ambiguous in some way, there will be multiple candidate substructures in the model. Pathfinder selects the correct one by analyzing the constraints inherent in the various relationship types used to structure the model, measuring for each candidate substructure the degree of semantic relatedness among the entities to which the query refers, and selecting the candidate which exhibits the highest degree of semantic relatedness.

Often, however, Pathfinder cannot select a single best candidate structure and a user must be the final arbiter. In such situations, the candidates must be presented to the user in a manner s/he is capable of understanding. In SystemX, this is done by generating English sentences that correspond to each candidate structure. The sentences currently generated are somewhat rough and halting. Research is planned which will improve their quality.

The database query module contains a component that translates the canonical semantic representations into a logical form. A second component translates the logical form into SQL. Different versions of this component would translate the logical form into other database languages.

SX – Nick Cercone, Paul McFetridge, Gary Hall, Dan Fass, Fred Popowich.

The SX system, like SystemX, is a natural language interface to a relational database which turns ordinary English questions into requests in SQL. SX is being developed as an advanced prototype executive information system for Rogers Cablesystems, a major national Canadian cable company, through a grant from the Canadian Cable Labs Fund. The initial target is a statistical database describing the customer service operations of the company, which counts sets of entities such as cable outages, telephone calls and customer service representatives, work orders, and pay methods.

Statistical databases are common in executive information systems, since executives are typically interested in summary information. Most conceptual modelling languages do not provide the facility to represent statistical concepts. Natural language queries to statistical databases often lack direct correspondence to database objects. For example, such queries often refer to the domain entities which are being summarized and these are only indirectly represented in the database via the statistics. This lack of correspondence means that the mediation of a conceptual model is even more necessary than in the general case. The SX group has focused on the proper representation of statistical concepts in such a model, and the use of such concepts in disambiguating natural language queries.

This extension into the domain of statistics is a major difference between SystemX and SX. The other major difference is that the natural language understanding module has been replaced in SX by a new one with a more modern grammar formalism and a new semantics. The grammar formalism is head-driven phrase structure grammar (see section 4). Two parsers have been developed, one written in Lisp by

McFetridge (McFetridge and Cercone 1990), and the other written in Prolog by Popowich and Vogel (1990, 1991). The two parsers are used to test competing ideas, which are sometimes easier to implement and test in one language than another. The semantics, developed by McFetridge (1991), is modelled on the structure of the Rogers database. The parser produces logical forms which are then passed to an adapted version of the database query module from SystemX.

Natural language front ends to deductive databases – Veronica Dahl.

Dahl has developed language front ends to her own deductive database and expert systems, which first introduced logic programming into the database and expert systems fields (Dahl, 1979a). These interfaces translate sentences in French or Spanish into a specially tailored typed three-valued logic system with three-branched quantification (Dahl, 1979b). through which they can detect failed presuppositions, disambiguate sentences and reject semantically as well as syntactically anomalous ones with little computational effort, and distinguish collective from distributive plurals to provide appropriate answers in each case. Semantic agreement is equated with syntactic well-formedness, which evens the relative costs of doing semantic versus syntactic tests – both can be performed through unification. These front ends have been adapted to other languages, such as Portuguese (by H. Coelho and Luis Pereira) and English (by D. Warren and F. Pereira). The three-valued logic into which these parsers translate English serves directly as a query language to the associated data base or expert systems, so the resulting code is extremely concise.

Another front end developed by Dahl and McCord (1983) provides in particular a logic grammar treatment of coordination in English with reconstitution of elided phrases. A prototype English parser being developed by Dahl, Fred Popowich and Michael Rochement will in a second phase be adapted for use as a database front end.

• **Speech recognition**

Most speech recognition work in NLP has been concerned with analyzing isolated words and phrases of about 100 words, with training sessions required for each speaker. This is much easier than recognizing *continuous speech*. Speech recognizers are needed for future telephone answering machines that can understand spoken commands, computerized telephone operators, and talkwriters that print what you say. Speech verification may be a component of tomorrow's security systems and might be used in future telephone answering machines that will recognize voices and deliver messages to particular callers.

Recognition of rapid speech – Tom Perry, Ross Saunders.

Perry and Saunders, both from Linguistics, are seeking to use linguistic ideas about speech segmentation for automated speech recognition. Perry is developing an inventory of half-syllables. No natural language contains more than about one hundred of these half-syllables. The number may be small enough for the rapid recognition of fast speech.

Saunders is investigating a new method for getting more information out of fricatives by examining the gradient of onset of the sound of a fricative. Fricatives are sounds made when two parts of the vocal tract come so close together that the air between them produces audible friction, e.g., the 'f-' in 'fin', 'v-' in 'van', 'th-' in 'thin' and 'th-' in 'this'.

Automated speaker identification – Binay Bhattacharya.

Bhattacharya has recently been working with Bell Northern Research (BNR) Labs in Ottawa on a speaker identification/verification system that can be trained to identify/verify people based on their voice patterns. The BNR system takes 36 voice recordings of a person. Each voice recording is decomposed into values on 15 parameters which are then time normalized into 32 frames, resulting in 32 by 15 numbers.

Bhattacharya has applied computational geometry techniques to organize the data into a geometrically partitioned space – a tree structure – and then has developed an efficient algorithm for traversing that tree, resulting in much faster speaker identification times than the methods BNR had been using (Bhattacharya et al, 1990).

• **Machine translation** – Veronica Dahl, Pierre Massicotte.

CWARC (the Canadian Workplace Automation Research Centre) has been developing a system called CRITTER for translating agricultural reports from French to English, and vice versa. Dahl and Massicotte have collaborated with CWARC in extending CRITTER's semantic coverage by incorporating

Dahl's methodologies for semantic verification which also provides better flexibility and better monitoring of operations. Under a research grant from IBM, Dahl studied the uses of static discontinuity grammars (see section 4) for generating conceptual graph representations of machine error messages (Brown, Pattabhiraman, et al, 1986), in view of their automatic translation. As an outgrowth of this research, Massicotte completed a linguistically principled system for generating conceptual graphs from functional structure representations (Massicotte, 1988).

4. Theory

This section is somewhat more technical than the preceding or following sections and can be skipped by the reader who is less interested in theoretical work. Section 5 does not assume knowledge of section 4. Some of the descriptions of theories presented below are adapted from Crystal (1991).

• Morphology

Pollard and Sag (1987) have proposed a program for organizing the lexicon hierarchically by lexical types. Lexical items inherit information which is not specific to them by according to their lexical type. In addition, relationships among words (for example, that between the 3rd person singular of the verb and its base) are expressed by lexical rules. Lexical rules may add information or transform information.

Paul McFetridge is initiating a project in information-based morphology to explore these proposals with particular focus on how lexical rules may be constrained and implementing a lexicon which is elaborated as needed rather than completely compiled when loaded.

• Syntax

Much of SFU's research in syntax aims to integrate computational and linguistic formalisms and ideas.

Static discontinuity grammar – Veronica Dahl, Fred Popowich, Susan Kindersley, Michael Rochemont.

Dahl has developed static discontinuity grammar (Dahl, 1989), which belongs to the family of discontinuous grammars also developed by Dahl, which are in turn a type of logic grammar. Logic grammars differ from traditional grammars in several respects including the form of their grammar symbols, which may include arguments representing trees or semantic interpretations, and the existence of processors based on specialized theorem-provers which give grammar rules a procedural aspect (Abramson and Dahl, 1989).

Discontinuous grammars can deal with many language phenomena involving discontinuity (i.e., the relating of constituents which may be widely separated in a sentence or phrase), for example unbounded dependency (see section 2) and free word order. Discontinuous grammars have provided the basis for related work in logic grammars (Matsumoto, 1988; Saint-Dizier, 1988; Sugimura et al, 1988) and in logic programming (Saint-Dizier, 1990).

The significance of static discontinuity grammar is that it transparently turns high level linguistic specifications into executable code. Such specifications can include two of the major tenets of modern linguistics: discontinuity, and the modular constraining of discontinuous constituents through various linguistic principles and filters, as in government-binding theory. Popowich has developed a left-corner parser of static discontinuity grammar. Dahl and Popowich (1990) describes the parser, parsing efficiency, and the tree admissibility interpretation of static discontinuity grammars,

Dahl and Abramson are developing static discontinuity grammar methodologies for top-down parsing of immediate dominance and linear precedence (ID/LP) rules and free word order rules with discontinuous constituents. Their methods are analogues for logic grammars of Shieber's extension to Earley's algorithm which parses grammars with free word order and ID/LP rules "efficiently" in the sense that the object context free grammar is not generated. These methods permit in particular the treatment of scrambling and ellipsis in languages such as Japanese.

Dahl has worked on merging government-binding theory into static discontinuity grammar (Dahl, 1988a). **Government-binding theory** – Veronica Dahl, Fred Popowich, Michael Rochemont Government-binding theory is a model of grammar descended from extended standard theory (Chomsky, 1982, 1986). Most of the descriptive research into the formal syntax of the different languages of the world

is within the government-binding formalism. The merging of one theory within the other is possible because the two have certain similarities: "both attempt to achieve maximal coverage with minimal expressive machinery, both can describe fairly free types of movement, both tend toward abstraction and generalization" (Abramson and Dahl, 1989, p. 153).

Static discontinuity grammars were successfully used for implementing government-binding theory in the system for machine error messages mentioned in section 3 (Brown, Dahl, et al, 1986). Dahl, Popowich and Dr. Michael Rochemont from the Linguistics department at the University of British Columbia are conducting further research into a static discontinuity grammar-based implementation of aspects of government-binding theory.

Unification-based grammars – Fred Popowich, Paul McFetridge, Carl Vogel.

Unification-based grammars are a family of grammars whose central operation is unification. Two descriptions can unify as long as they do not contain conflicting information. The result contains all the information that is in either description.

Unification-based grammars are declarative and lexical. Their declarative nature makes them relatively easy to understand and modify, since a grammar states relationships instead of giving procedures for their computation. Their lexical nature reduces grammar development to lexicon development because syntactic classes contain information about the classes with which they combine, which means that only a small number of grammar rules and principles are needed, and that those rules and principles are schematic and fixed. However the drawback is large lexicons with complex lexical entries. To avoid redundancy and to capture linguistic generalizations in such lexicons, lexical types and lexical rules have been introduced. Lexical types eliminate "vertical redundancy" by factoring out general information shared by whole syntactic classes, organizing lexical entries into multiple hierarchies based on the information they share, and inheriting down general information. Lexical rules eliminate "horizontal redundancy" by identifying lexical entries that have common, specific patterns which cannot be inherited via lexical types. In such cases, a lexical rule fills in the specific information.

Head-driven phrase structure grammar – Paul McFetridge, Fred Popowich, Carl Vogel.

Head-driven phrase structure grammar (Pollard and Sag, 1987) is one of the best known unification-based formalisms. A distinctive feature of the grammar is that attribute-value matrices are used to represent lexical entries and grammar rules and principles. A second distinctive feature is its head-driven nature: the head constituent of a phrase is a central notion. Otherwise, the grammar borrows freely from previous unification-based grammars. The treatment of syntactic categories, syntactic features, and some of the principles are from generalized phrase structure grammar (Gazdar et al, 1985). A number of the lexical rules are similar to those in lexical-functional grammar (Kaplan and Bresnan, 1982).

Chart parsing implementations of head-driven phrase structure grammar have been developed in Lisp by Paul McFetridge and in Prolog by Fred Popowich and Carl Vogel. Chart parsing is a type of parsing in which all syntactic structures which are built are placed on a single graph structure called a chart. A successful parse of a sentence is a list of edges and nodes that includes the start node of the chart, all the words in the sentence, and the end node.

Tree unification grammar – Fred Popowich, Susan Kindersley.

Tree unification grammar (Popowich, 1989a, 1989b) resembles head-driven phrase structure grammar in certain respects. Both formalisms use signs as information structures and provide facilities for establishing the relationships between signs. Tree unification grammar incorporates a greater degree of lexicalization, using one rule as opposed to the four or five rules described in Pollard and Sag (1987, forthcoming). Head-driven phrase structure grammar also relies on the use of universal grammar principles which apply in conjunction with the grammar rules. Many of these principles make use of functionally dependent values: they contain calls to relatively complex functions like *collect-indices*, *selectively-combine-semantics*, *order-constituents* and *bip* (application of the binding inheritance principle). In tree unification grammar, the relationships between the information contained in different signs is stated in terms of unification, or very simple list operations, within the lexical templates.

Tree unification grammar also differs from head-driven phrase structure grammar in the kind of derivational structures that it allows. In tree unification grammar the structures are binary, consisting of a specified functor and argument. For the structures used by head-driven phrase structure grammar, which are called phrasal signs, there is no theoretical limit on the branching factor. In the phrasal signs of head-driven phrase structure grammar, it is very awkward to describe specific relationships relevant to constraints on reflexive pronouns. Tree unification grammar specifies three different reflexive relationships contained in its *R-GPR*, *R-PR* and *R-NPR* lexical templates (Popowich, 1989a). The different relationships between semantic and reflexive information can be captured more elegantly in the formulation of head-driven phrase structure grammar introduced in Pollard (1989), but it is still not as concise as the tree unification grammar characterization that we have described.

A parser has been written that can use tree unification grammar in an NLP system (Popowich, 1989b). The prototype for this system will form the basis of a larger system for developing and testing natural language grammars that can be used in a natural language interfaces. The larger system is currently being developed by Susan Kindersley (Popowich and Kindersley, 1991).

Parsing as inheritance reasoning – Carl Vogel, Fred Popowich, Nick Cercone.

Vogel and Popowich (1990) have proposed that parsing with a unification-based grammar can be viewed as reasoning over an inheritance network. This research is of a system-building nature in comparison to the related research of Nebel and Smolka (1989) who explore the relationship between terminological reasoning systems and unification grammars with respect to the computational properties of computing subsumption in either system.

The aim of the research is to explore the extent of the relationship which begins with a number of similarities observed between unification-based grammar and inheritance reasoning. First, unification and inheritance reasoning are closely related: the monotonic unification operator can directly model strict inheritance, and extended unification algorithms can be used to describe defeasible or mixed strict and defeasible inheritance (Bouma, 1990).

Second, lexical types and inheritance reasoning are closely related: lexical types allow common information to be shared by lexical entries, while inheritance has frequently been used in NLP to allow the sharing of information in the lexicon (Flickinger et al, 1985; Evans and Gazdar, 1990). Lexical information is organized in an inheritance hierarchy and the information associated with a specific lexical entry is determined by inheritance over this hierarchy. Information relating to grammar rules and principles can also be captured in a similar manner (Pollard, 1985; Fraser and Hudson, 1990). The construction of an inheritance hierarchy allows the concise statement of linguistic information, be it lexical, phonological, morphological, syntactic or semantic.

Third, inheritance need not be restricted to descriptive hierarchies such as a hierarchy of lexical types, but can also be used to establish relationships in derivational structures such as parse trees. Inheritance can thus be used in parsing and in generation.

The result of this research is a pedagogical tool for describing the structure and application of head-driven phrase structure grammar and, produced as an artifact, a cleanly specified – if inefficient – alternative parsing strategy. More efficient parsers for head-driven phrase structure grammar do exist: Proudian and Pollard (1985) developed a chart parser for an early version of the formalism, and there are McFetridge's and Popowich's chart parsers too. A nice side effect of the inheritance reasoner is that it provides an elegant way to represent and process unknown words.

Grammar rule induction in connectionist networks – Robert Hadley.

A serious drawback of connectionist systems is that they do not permit the fast integration of new vocabulary and linguistic patterns which children exhibit once they have learned rudimentary syntax. The aim of this research to get a connectionist network to learn, on its own, some basic rules of syntax for natural language. These rules will be acquired as patterns by the connectionist network.

It is known that when semantic information is incorporated into the learning process, classical algorithmic methods for inducing syntactic patterns can be made significantly more efficient. We are integrating the semantic acquisition process with the induction of syntactic patterns on a connectionist network, so that the acquisition of syntactic patterns will involve, as a sub-process, the acquisition of a small hierarchy

of concepts, beginning with categories corresponding to objects, actions, and object-modifiers. We have closely examined the problem of how primitive words (which denote objects, actions, and object-modifiers) become associated on a connectionist network with the real world objects they actually denote. There have been past designs of connectionist architectures for discovering this "denotation relationship" which obtains between particular words and particular aspects of a situation, but these designs have adopted the unrealistic assumption that the learner (say, a child) already knows precisely what "state of affairs" a sentence is intended to describe.

In addition, Hadley's recent research has included a critical examination of connectionists systems which are purported to exhibit the compositionality and systematicity characteristic of natural language. I argue that many of these claims are seriously exaggerated (Hadley, 1990).

• Semantics

A representation for NLP using extended semantic networks – Nick Cercone.

Cercone et al's extended semantic network representation (Cercone et al, to appear) can represent states, events, actions, logical and natural language quantifiers, expressible intentionality, modalities, adverbials, comparatives (explicit and implicit), and the meanings of complex concepts such as walking. The concept representation is hierarchical and recursive rather than dependent upon primitives. Formal methods for interpretation and manipulation of the network constructs have been developed and implemented. Complex concepts are efficiently processed. Lambda-abstracted predications are uniformly handled in the representation so that sentences such as "loving ones neighbours is a virtue" are understood.

Work has concentrated on two main problems: modifiers, and inferencing and reasoning strategies. Iterated modifier analysis is used advantageously to account for comparative and descriptive modifiers. In an implementation, we have refined our treatment of English modifiers.

A means of promoting inferences is being developed through the use of supra-organizations which will be imposed on the network database to allow rapid access to facts contextually pertinent to concepts.

A stratified meaning representation for natural language discourse – Tomek Strzalkowski, Nick Cercone.

The stratified model, developed by Strzalkowski (1986) uses a lambda-categorial language for meaning representation. The model applies a series of disambiguating transformations to a discourse fragment before assigning it a final representation. The transformations include morphological analysis, lexical ambiguity resolution, syntactic parsing, computation of extra-sentential dependencies, and pragmatic evaluation in discourse context. The computation of extra-sentential dependencies is performed on an intermediate, formal representation of the discourse fragment. The transformations are done using Montague's (Thomason, 1974) style of coupling syntactic and semantic processing but not his use of an intensional logic – the intermediate representation used is a lambda-categorial language (a typed predicate calculus language with a lambda-operator). The final representation of discourse content, while not decided upon, will have a form resembling that of "abstract situation" from Barwise and Perry's (1983) situation semantics. The model has been used to investigate two discourse problems: first, the resolution of anaphoric and cataphoric reference across sentences (Strzalkowski and Cercone, 1986), and second, the representation and manipulation of non-singular concepts (Strzalkowski and Cercone, 1989). Presently, there is no implementation of the stratified model.

The resolution of cross-sentence anaphoric and cataphoric reference. This work has so far been restricted to cases of anaphora in two sentence stories. In these stories, cases have been selected where a reference, if it can be computed at all, has a unique antecedent. A set of rules has been formalized for computing the references in these cases, and the transformation which encompasses these rules, and the rules themselves, have been related to the stratified model.

Cross-sentence cataphoric references have also been studied and some solutions specified for certain cases. Another discourse problem that has been studied using the same methods is repairing references in discourse. An example of reference repair is given in the following three sentence discourse: "I bought a black coffee in the pub. I brought it back to my office to drink. The cream in my office was off and it (the coffee) tasted bad."

The representation and manipulation of non-singular concepts. This work (Strzalkowski and Cercone, 1989) generalizes the treatment of the concepts given in Strzalkowski and Cercone (1986). Non-singular concepts are abstract entities that embrace a variety of smaller or larger collections of instances. They are usually referred to using bare plural noun phrases (such as 'birds', 'alligators' and 'presidents'), definite singular noun phrases with a generic interpretation ("the alligator," "the president"), mass nouns ('water', 'gold'), and functional uses of definite descriptions ("the number of students," "the temperature").

Non-singular definite descriptions appear in a number of reasoning puzzles, for example, the following syllogism adapted from Partee (1972):

The president is elected every four years.

The president is Bush.

Thus, Bush is elected every four years.

This erroneous conclusion arises because we assign a non-singular interpretation to "The president" instead of a singular one.

Our solution is a representation based on a partially ordered structure of levels in which the objects of the same relative singularity are assigned to the same level. Our choice of the representation has been motivated by the following main concerns: (1) the representation should systematically distinguish between those language terms that are used to refer to objects of different singularity, that is, those classified within different but related levels of the model; (2) the representation should capture certain types of inter-sentential dependencies in discourse, most notably anaphoric-type cohesive links; (3) finally, the representation should serve as a basis for defining a formal semantics of discourse paragraphs that would allow for capturing the exact truth conditions of sentences involving non-singular terms, and for computing inter-level inferences. Some initial progress has been made on (1) and (2) but (3) is currently under investigation. We believe that our approach promotes computational feasibility, because we avoid the identification of general terms like 'temperature', 'water', etc., with intensions, that is, functions over possible worlds. In our theory, the concept of non-singularity has a local (often subjective) character.

The met* method of metaphor and metonymy interpretation – Dan Fass.

The met* method (Fass, 1988, 1991a, 1991b) distinguishes selected examples of metonymy from metaphor and from literalness and anomaly in short English sentences. Consider, for example, a waitress in a restaurant saying "The ham sandwich is waiting for his check." This sentence contains a metonymy while "The car drank gasoline" contains a metaphor. The difference between a metonymy and a metaphor, according to Lakoff and Johnson (1980), is that in metonymy one entity stands for another whereas in metaphor one entity is viewed as resembling another. This difference can be seen in the above sentences: "The ham sandwich" stands for the male person who ordered the ham sandwich, while "The car" is viewed as resembling an animate drinker in that both use up a liquid, take in the liquid through an aperture, etc.

In the met* method, literalness is distinguished because it satisfies contextual constraints that the non-literal others all violate. Metonymy is discriminated from metaphor and anomaly in a way that supports Lakoff and Johnson's (1980) view of metonymy and metaphor, permits chains of metonymies (Reddy, 1979), and allows metonymies to co-occur with instances of either literalness, metaphor, or anomaly. Metaphor is distinguished from anomaly because the former contains a relevant analogy, unlike the latter. The met* method is part of collative semantics (Fass, 1989, 1990), a semantics for natural language processing, and has been implemented in a computer program called meta5. Collative semantics is also concerned with the representation and resolution of lexical ambiguity. The meta5 program contains a lexicon of 500 word senses, a small grammar, and routines containing collative semantics.

Salience in language generation – T. Pattabhiraman, Nick Cercone.

The role of salience in decisions in natural language generation has been examined. The salience of an entity in natural language generation represents its intrinsic prominence. It is therefore used in several decisions in natural language generation as a measure of preference that justifies the choices committed. Salience is a multi-aspect notion that includes vividness and topicality. The interactions among salience-

influenced decisions can be characterized as synergistic, competitive, complementary, asynchronous, concurrent and weighted. The problems involved in making these interactions operational in natural language generation systems are being examined, in particular, from the viewpoint of efficient, maximally-deterministic generation. Computational mechanisms are being developed to capture the influences of salience in natural language generation decisions such as content selection, lexical selection, tactical linearization and the generation of pronouns (Pattabhiraman and Cercone, 1990a, 1991).

Pattabhiraman is also interested in the problem of evaluating theories and systems in natural language generation (Pattabhiraman and Cercone, 1990b) and is currently working on developing a coherent classification of natural language generation systems, and spelling out a set of evaluation criteria that are relevant to each kind. The classification of natural language generation systems is based on a number of dimensions such as objectives, stages of evolution and approaches to the modelling of linguistic knowledge.

• Pragmatics

SFU's work in pragmatics has been concerned with dialogue between a natural language interface and a user when one of them has faulty or missing information, specifically, how to keep the dialogue going and how to repair the dialogue. This work can also be viewed as introducing robustness into NLP systems because they attempt to keep a dialogue going in situations where it has broken down. The main problem in this work, which has been overlooked by many other researchers, is fault finding from the dialogue: is the faulty or missing information in the database of the natural language interface or with the user?

Predicting and explaining query failure – Stefan Joseph, Romas Aleliunas, Nick Cercone.

Database systems don't contain all the information needed to model their target domain. Users' questions usually contain assumptions about the domain. A null answer arises when or more assumptions of a question are incorrect. For example, "which graduate students have taught CMPT681?" (example due to Bonnie Webber) assumes that graduate students can teach that course or "which text book is used for CMPT898?" assumes that CMPT898 is a course on which a text book is used. A null response – "no" – is not very helpful. What is needed is more informative responses.

Joseph has developed a way of predicting null responses to database queries in a way that avoids wasteful search of secondary memory and a way of generating meaningful explanations of why no result was found (Joseph and Aleliunas, 1991). A distinction is made between two main kinds of null value: "property is inapplicable" as in a male person's maiden name (which violates an obvious semantic constraint on male persons), and "value at present unknown." This research developed from Kao's work (Kao et al, 1988; Cercone et al, 1990) which analyzed the reasons for the null responses once the database access resulted in a no response.

Joseph uses an inference procedure to detect empty responses to queries and to give a reason why a response was empty. The inference procedure uses three sources of information: the query itself (an expression in relational calculus) , the RM/T* catalogue which describes the structural and functional dependencies of the actual database, and a knowledge base whose rules are semantic constraints on the database. There is a general rule of inference for each operation of relational calculus (e.g., join, projection, selection). As soon as the inference procedure determines that one of the attribute-domain sets is empty, further database searching is unnecessary and the query must return an empty response. The inference procedure's actions are strictly controlled by the structure of the query.

A belief-based view of ill-formed input – Dan Fass, Gary Hall, Nick Cercone, Chris Groeneboer, Paul McFetridge, Fred Popowich.

Ill-formed input refers to "lexical problems such as misspelled words, sentential problems such as missing words or phrases and bad word order, semantic problems such as anomalous or self contradictory sentences, and contextual problems such as incoherent requests or continuations" (Allen, 1983). Users of natural language interfaces produce ill-formed input which must be handled by the interfaces.

Perhaps the biggest problem with ill-formed input is locating it. We have tackled this problem by treating ill-formed input as a clash of beliefs between a user and an interface: the ill-formed input is then viewed as the false or incorrect beliefs of either the user or interface. To see how this works, consider

what happens when a natural language interface parses a sentence containing the item 'recieve'. 'Recieve' is not in the interface's lexicon, hence there is a conflict of belief. There are two obvious causes of the conflict: first, 'recieve' is a spelling error by the user (i.e., the user has *false* beliefs about the spelling of 'receive'), and second, 'recieve' is an unknown word to the interface (i.e., the interface has *missing* beliefs about the word 'recieve').

This central idea about clashing beliefs has been incorporated into a framework we have developed for treating ill-formed input in which an interface and its users each has a "belief model" about itself and other participants with which it engages in dialogue (Fass et al, 1990; Fass and Hall, 1990). Each model contains beliefs of that participant about its language and its knowledge of the domain. The clashes of beliefs lead to a far more extensive classification of situation in which ill-formed input occurs than previous approaches to ill-formed input.

5. Integrated Systems

The aim in producing integrated systems is to combine natural language with logic programming, general-purpose interfaces, databases, and expert/reasoning systems.

• **Logic programming** – Veronica Dahl, Fred Popowich, Susan Kindersley, Greg Sidebottom, Joerg Ueberla.

Veronica Dahl has been investigating the translation of high-level linguistic specifications in static discontinuity grammar into executable code in logic programming (Dahl et al, 1986). An approach to concept type hierarchies in logic programming has also been developed (Massicotte and Dahl, 1988) which, while reasonably efficient, is more portable than prior approaches. Together with Greg Sidebottom and Joerg Ueberla, Dahl is also developing constrained logic programming techniques which can provide a general, easy-to-use way of describing design worlds for efficient automatic problem-solving. These techniques combine a variation of Freuder's algorithm for synthesizing constraint expressions with a variation of Sidebottom's (1991) integer constraint satisfaction methods, and incorporates heuristic selection functions which are appropriate for design problems in general.

• **Grammar development tools**

The grammars used in NLP systems are becoming very sophisticated. Developing such grammars requires designing, testing and modifying grammar rules, principles and lexicons. Tools for developing such grammars are needed by system developers (and sophisticated users) of applications like natural language interfaces to databases, and by students and researchers for understanding existing grammars, for extending them, and for developing new grammars. Tools will make it easier to study how linguistic problems are handled by different grammars, to import ideas from one grammar into another, and to describe fresh linguistic problems within an existing grammar.

TREETOOL, a tool for displaying tree structures – Fred Popowich, Sue Baker, Rob Hamm.

A graphics module for displaying the tree structures associated with tree unification grammar has been developed on SUN workstations (Baker et al, 1990), and is being ported to X-Windows. The module has also been incorporated into a government-binding theory parser and Lisp-based and Prolog-based head-driven phrase structure grammar parsers. The module is a useful tool for any NLP system for displaying complete parse trees and partial analyses.

Toolkit for developing unification-based grammars – Fred Popowich, Susan Kindersley.

Development tools exist for many general purpose unification-based grammar formalisms like PATR-II (Shieber et al, 1983), D-PATR (Kartunen, 1986), and STUF (Uszkoreit, 1986; Bouma et al, 1988). Popowich and Kindersley are interested in providing a system for implementing particular unification-based grammars like tree unification grammar and head-driven phrase structure grammar which have specific development problems not found in the more general grammars. For instance, head-driven phrase structure grammar allows functions to appear as feature values, and tree unification grammar possesses relations requiring a element to be an arbitrary element of a list. These operations are not available in PATR or in STUF.

The toolkit will make it easier to compare particular grammars. For example, the toolkit will facilitate the examination of recent proposals for the treatment of anaphora, coordination, comparatives and

adjuncts in head-driven phrase structure grammar (Popowich, 1989a, 1989b) and the consideration of alternatives.

The grammar development tools will be developed by Susan Kindersley within the ProWindows environment of Quintus Prolog. Since unification is a primitive operation in logic programming frameworks, it is appropriate to adopt a logic programming language like Prolog to implement the system.

The LINGKIT system for developing government-binding theory grammars – Veronica Dahl, Fred Popowich, Michael Rochemont.

Many of the world's languages have been described within government-binding theory. Since the theory is still evolving, it is important for linguists to be able to design grammars and principles and to examine the correlation between their hypotheses and the natural language data. The LINGKIT system will allow linguists to focus on the treatment of unbounded dependency constructions. The system is based on a declarative high level logic grammar formalism in which unbounded movement can be easily described, namely static discontinuity grammar (see section 4).

• **General-purpose interfaces** – Chris Groeneboer, Nick Cercone.

Users prefer different modalities at different stages of learning to use a system and for different applications. Naive users learn to use new systems faster with natural language, while experienced users prefer control keys because they are faster (Napier et al, 1989). Graphical displays may be preferred for certain tasks, e.g., clicking on a region of a map to get information about that region. What is needed are user interfaces that can handle different modalities and different applications.

Chris Groeneboer and Nick Cercone (1991) have developed a modular architecture for a multi-modal, multi-application user interface. The architecture has five layers consisting of: user, modalities, the interface itself, applications, and the underlying operating system. Requests and responses undergo transformations as they pass through the layers. User requests pass down through the layers to the appropriate application. System responses pass up through the layers to the user in an appropriate modality for presentation. The layered architecture is designed to make changes easy, for example, adding a new modality, application or user. Control resides with the interface itself which contains a set of experts: a modality expert, an applications expert, a representation expert, an interaction expert, and a coordinator which organizes the other experts. Each expert has access to two knowledge bases specific to its expertise, one that contains general domain-independent information, and another that contains specific domain-dependent information. For example, the interaction expert has a domain-independent knowledge base about human-computer interaction in general while the domain-dependent knowledge base contains user models which in turn contain information about specific users.

Groeneboer and Cercone have also developed an underlying philosophy of interface design based on the constructivist paradigm. Within this paradigm, knowledge is viewed as constructed by a user to fit with the user's experiences of the world. Attention is paid to the processes by which users organize their experience. The significance of this paradigm for interface design is that attention is paid to how users construct the interface environments they want.

Other researchers whose work is helping is Tom Calvert, the Intelligent Graphic Interface (IGI) Group, and Gerri Sinclair's EXCITE Lab.

• **Database systems** – Veronica Dahl, Jai-Wei Han, Gary Hall, Ranabir Gupta.

Dahl has worked on integrating NLP and database techniques (Dahl, 1982, 1990).

She has developed a typing system that reduces many type checking operations to unification and which have found applications in deductive databases (Dahl, 1991). The distinctive feature in Dahl's work on deductive relational database systems is the use of logic as a single tool for formalizing and implementing different aspects of database systems in a uniform manner. The use of logic throughout allows for a uniform representation of programs, parser, data, inference, semantic interpretation and query evaluation. Its use both as the theoretical framework and as the implementation means allows for a great degree of data-independence and non-procedurality. Dahl has developed a single logical formalism, serving both as a meaning representation language and a query language (see section 3), which contains features of logic programming database querying and defining and features of representations of natural language.

Han has been working on constraint-based reasoning in deductive databases (Han, 1991). Constraint-based reasoning has been studied extensively in logic programming and artificial intelligence, for example, the Echidna reasoning system described below. Han has developed a constraint-based reasoning technique for deductive databases involving functional linear recursions. The application studied is booking air flight tickets.

Hall and Gupta (1991; Gupta and Hall, 1991) have been studying how to integrate the representation of the static and dynamic aspects of a domain into a single conceptual model. Hall is looking to extend Pathfinder (used in SystemX and SX) to measure semantic relatedness in models which include relationship types that Pathfinder cannot presently manage.

Generating a conceptual model is a major phase of customizing a natural language interface to a database. Hall has been looking into means to automate this phase as much as possible. A related problem is how to update changes which occur in the domain of the database to a natural language interface. Successful automatic generation techniques should ease this problem because the techniques should be able to deduce changes in the domain which are prompting changes to the logical model and to the patterns of data in the database itself.

• **Expert systems** – Veronica Dahl, Fred Popowich, Paul McFetridge, CSS Expert Systems Laboratory, Joerg Ueberla.

Veronica Dahl (1986, 1990) has proposed an integrating framework for the development of expert systems that either have linguistic knowledge as their domain of expertise, or use it in their interfaces. Also, as mentioned earlier in section 5, she is presently working on an expert system for automatic design problems, which provides in particular some built-in heuristics that are applicable to all automatic design domains, plus modular definition facilities for completing the heuristic information with problem specific data for each application. A prototype is being developed with Greg Sidebottom and Joerg Ueberla for the specific problem of designing computer configurations satisfying a user's particular needs (Dahl et al, 1991).

Some NLP researchers are using an expert system reasoning engine for NLP applications. The reasoning engine is called Echidna developed by Bill Havens, Director of the CSS Expert Systems Laboratory, and his research team. Fred Popowich and Bill Havens are writing a natural language parser in Echidna, and Paul McFetridge is using Echidna to develop a decomposition of task verbs for a robot assembly system.

Echidna is a constraint logic programming system for model-based expert systems applications. It improves on existing expert systems by combining aspects of predicate schema-based object-oriented knowledge representations, dataflow dependency backtracking, and constraint logic programming (Havens et al, 1990a). It includes a justification-based clausal reasoning maintenance system. Echidna is the first logic programming language that integrates active constraint reasoning techniques to detect failure in search sooner and dependency backtracking to avoid unnecessary computation when failure is detected (Havens, 1990).

A prototype, Echidna version 0, has been in use internally since April 1990 (Havens et al, 1990b). Testing began on an internal release of Echidna version 1 in February 1991. This version is implemented in C++, has an interface in X-windows and runs on Sun and NeXT computers. In the future, it is planned to provide Echidna on other platforms with support for distributed systems and hypothetical reasoning. Echidna will be made available commercially in the future.

Prototype knowledge bases are being implemented in the area of mechanical and architectural design. Also being implemented is further support for problems in diagnosis and problems in operations research (such as graph colouring and scheduling). Echidna version 0 is also being evaluated on a number of applications, including an intelligent computer-aided design system called *intelCAD*, an intelligent graphics interface project for supervisory control called *IGI*, and an automotive diagnosis project for engine computer control systems called *SuperEAS*.

Appendices

A. Personnel

• Faculty

Principal researchers – text processing

Nick Cercone –Computing Science; Director, Centre for Systems Science

Veronica Dahl – Computing Science; Director, Logic Programming Laboratory

Paul McFetridge – Linguistics; Director, Natural Language Laboratory

Fred Popowich – Computing Science

Principal researchers – speech processing

Binay Bhattacharya – Computing Science

Tom Perry – Linguistics

Ross Saunders – Linguistics

Associate researchers

Robert Hadley – Computing Science

Jai-Wei Han – Computing Science

Bill Havens – Computing Science; Director, Expert Systems Laboratory

Wo-Shun Luk – Computing Science

Michael Rochemont – Linguistics, University of British Columbia

• Staff

Research Scientists

Dan Fass

Chris Groeneboer

Gary Hall

Susan Kindersley

• Students

Present graduate students

Ranabir Gupta – PhD in Computing Science (a framework to support object-oriented views of dynamic database domains)

Chantal Lunardi – PhD under Special Arrangement (multi-ethnic characteristics of computer enrichment)

T. Pattabhiraman – PhD in Computing Science (salience in natural language generation)

Allan Bennett-Brown – MSc in Computing Science (neural network model of early speech perception)

Xiobing Chen – MSc in Computing Science (extracting functional dependencies and finding synonyms)

Alison Marchant – MA in Linguistics (subject needed)

Michael McGuire – MSc in Engineering (speech recognition using neural networks)

Bruce Wiebe – MSc in Computing Science (computational phonology)

Eric Wu – MSc in Computing Science (a benchmark method for SQL query performance)

Recent graduate students

Yandong Cai – (1990) MSc in Computing Science (attribute oriented induction in relational databases)

Carl Vogel – (1990) MSc in Computing Science (inheritance reasoning and head driven phrase structure parsing)

Mimi Kao – (1988) MSc in Computing Science (generating quality responses from null answers to queries)

Stefan Joseph – (1988) MSc in Computing Science (eliminating unnecessary database accesses with natural language interfaces)

Pierre Massicotte – (1988) MSc in Computing Science (generating conceptual graphs from functional structures)

B. Hardware and Software

The following is developed from Barry Shell's leaflet *CSS Natural Language Laboratories*. The Natural Language Interface Laboratory and Logic Programming Laboratory are equipped with Macintoshes, NeXTs, PCs and SPARCstation 1 workstations. Software available includes the Oracle Database, the G2 expert system shell, Common Lisp, Quintus Prolog, ALS Prolog, Sicstus Prolog, Arity Prolog, and LPA Prolog.

C. The Centre for Systems Science

The CSS was established at SFU in 1987 as a provincial Centre of Excellence to foster basic cooperative research and education in advanced systems including artificial intelligence, robotics, computing science, and microelectronics. The Director of the CSS from its inception has been Nick Cercone. The CSS supports three broad research groups in: intelligent systems, computer & communication systems, and microelectronics. These research groups involve approximately 84 faculty, drawn principally from the Schools of Computing Science and Engineering Science, but also including faculty from Linguistics, Mathematics, Physics, Geography, Philosophy, Psychology, Communication, Education, Kinesiology, and Business Administration.

Several research laboratories have been established within the CSS, including the Natural Language Laboratory (Director: Paul McFetridge), the Logic Programming Laboratory (Director: Veronica Dahl), and the Expert Systems Laboratory (Director: Bill Havens). Work has from all three of these laboratories has been described in this report.

The CSS has established a research support computing network which currently serves over 400 users in twelve departments. The network consists of over 400 computers, and growing. Computers and related equipment have come from a variety of sources, the major ones being faculty research grants, the CSS, and SFU.

The CSS has engaged in numerous other activities, including a University/Industry joint appointments option for hiring faculty, and several research support programs, including the *research grants* competition and *support for research groups* competition. These programs have been replaced by two newer programs, the *matching grants policy* and *strategic initiatives program*. Matching grants are designed to provide a four or five to one pay back to SFU for funds invested. The next appendix lists the major grants won by the main researchers in NLP at SFU.

D. Grants of Principal Researchers in Natural Language Processing (1986 on)

As many acronyms are used in the tables that follow, a key is supplied.

Key:

AGAR	=	Assistance Grants for Applied Research (program of BCSC).
ASI	=	Advanced Systems Institute of British Columbia, Burnaby, BC.
BCSC	=	British Columbia Science Council, Metrotown, Burnaby, BC.
BNR	=	Bell Northern Research, Ottawa. ON.
CCLF	=	Canadian Cable Labs Fund, Metrotown, Burnaby, BC.
CIAR	=	Canadian Institute of Advanced Research, Toronto, ON.
CSS	=	Centre for Systems Science (part of SFU).
CWARC	=	Canadian Workplace Automation Research Centre, Montreal, PQ.

IBM	=	International Business Machines, Toronto, ON.
IRIS	=	Institute of Robotics and Intelligent Systems, Ottawa, ON
ISE	=	International Submarine Engineering, Port Coquitlam, BC.
MART	=	Market Assessment (program of BCSC).
NCE	=	National Centres of Excellence (with NSERC), Ottawa, ON.
NRCC	=	National Research Council of Canada, Ottawa, ON.
NSERC	=	National Science and Engineering Research Council, Ottawa, ON.
PRECARN	=	Precompetitive and Applied Research Network, Ottawa, ON.
SFU	=	Simon Fraser University, Burnaby, BC.
STDF	=	Science and Technology Development Fund (program of BCSC).
STF	=	Science and Technology Fund (program of BCSC, replaces STDF and AGAR).
SUR	=	Shared University Research (program of IBM).
UVIC	=	University of Victoria, Victoria, BC.

• Academic grants

Amount	Period	Source	Recipient(s)	Notes
141,000	93-94	IRIS B-4 node	Cercone et al	Design and human interfaces (4 years)
139,000	92-93	IRIS B-4 node	Cercone et al	
146,000	91-92	IRIS B-4 node	Cercone et al	
132,000	90-91	IRIS B-4 node	Cercone et al	
36,611	91-92	NSERC operating award	Cercone	Knowledge representation and natural language understanding (6 years)
36,611	90-91	NSERC operating award	Cercone	
36,611	89-90	NSERC operating award	Cercone	
34,300	88-89	NSERC operating award	Cercone	
34,300	87-88	NSERC operating award	Cercone	
28,480	86-87	NSERC operating award	Cercone	
12,500	91-92	CIAR, IRIS	Cercone	CIAR Associate
12,500	91-92	PRECARN	Cercone	IRIS Area Coordinator (for teaching release)
175,000	84-91	NRCC	Cercone	*Computational Intelligence* journal
30,000	90-91	CSS matching	Cercone	
107,880	88-89	NSERC strategic grant	Cercone, Luk	Accessing information through ordinary language (3 years)
110,380	87-88	NSERC strategic grant	Cercone, Luk	
107,880	86-87	NSERC strategic grant	Cercone, Luk	
21,000	86-87	CSS research grant	Cercone, Luk	SystemX
36,500	86-87	UVIC President's research grant	Cercone	Canadian 5th Generation Research Society
33,490	89-90	CWARC collaboration	Dahl	Automatic translation of agricultural reports
11,845	88-89	CWARC	Dahl	Automatic translation of agricultural reports
25,187	91-93	NSERC operating award	Dahl	Logic applied to natural language processing and expert database systems
25,187	90-91	NSERC operating award	Dahl	
25,187	89-90	NSERC operating award	Dahl	

25,187	88-89	NSERC operating award	Dahl	
25,187	87-88	NSERC operating award	Dahl	
25,187	86-87	NSERC operating award	Dahl	
5,000	88-89	CSS research grant	Dahl	automatic translation of agricultural reports
36,000	91-94	NSERC operating award	Popowich	unification-based grammars
30,000	89-91	NSERC operating award	Popowich	unification-based grammars
105,000	89-92	ASI fellowship	Popowich	unification-based grammars
10,000	91-92	Bridge/CSS strategic	Perry	speech recognition
5,000	88-89	CSS research grant	Perry	speech recognition
3,000	87-88	CSS research grant	Perry	speech recognition

• Industrial contracts

Amount	Period	Source	Recipient(s)	Notes
20,000	91-92	BNR	Bhattacharya	Speaker identification/verification
20,000	90-91	BNR	Bhattacharya	Speaker identification/verification
100,000	92-93	Rogers Cablevision	Cercone et al	Expert information management (3 years)
100,000	91-92	Rogers Cablevision	Cercone et al	
83,640	90-91	Rogers Cablevision	Cercone et al	
18,000	90-91	BCSC MART grant	Cercone	Market assessment of SystemX competition
187,000	86-87	IBM SUR grant	Dahl	machine translation
??	91-92	International Sub-marine Engineering	McFetridge	Task verbs for robot control

• Infrastructure and equipment grants

Amount	Period	Source	Recipient(s)	Notes
75,000	93-94	NSERC infrastructure	Cercone, Kameda, Hafer	
75,000	92-93	NSERC infrastructure	Cercone, Kameda, Hafer	
75,000	91-92	NSERC infrastructure	Cercone, Kameda, Hafer	
45,055	93-94	Ministry of Advanced	Cercone	For IRIS B-4 "Design and human
45,055	92-93	Education & Job Training	Cercone	interfaces" 90-94 (4 years)
45,055	91-92	provincial match	Cercone	
10,000	90-91		Cercone	
86,000	90-91	NSERC infrastructure	Cercone, Kameda, Hafer	
86,000	89-90	NSERC infrastructure	Cercone, Kameda, Hafer	
86,000	88-89	NSERC infrastructure	Cercone, Kameda, Hafer	
60,480	87-88	NSERC infrastructure	Cercone, Kameda, Hafer	
60,480	86-87	NSERC infrastructure	Cercone, Kameda, Hafer	

41,557	89-90	NSERC equipment	Cercone et al	
13,852	89-90	CSS matching	Cercone et al	For NSERC equipment 89-90
83,700	87-88	SUN Microsystems equipment grant	Cercone	Support of NSERC strategic grant 86-89
69,768	87-88	NSERC equipment	Cercone, Kameda, et al	

References

Abramson, Harvey, and Veronica Dahl (1989). *Logic Grammars*. New York, NY: Springer Verlag.

Allen, James F. (1983). Notes from the Editor. *American Journal of Computational Linguistics*, Special Issue on Ill-Formed Input, 9, (3-4).

Baker, Sue, Rob Hamm, and Fred Popowich (1990). The TREETOOL User's Manual. Technical report CMPT TR 90-09, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Barwise, John, and John Perry (1983). *Situations and Attitudes*. Cambridge, MA: MIT Press.

Bhattacharya, Binay K., J. Chiabaut, D. Graf, D. Peacocke, and D. Shibahara (1990). Text-dependent speaker identification using telephone speech. *Proceedings of Canadian Conference on Electrical and Computer Engineering*.

Bouma, Gosse, Ester König, & Hans Uszkoreit (1988). A Flexible Graph-Unification Formalism and its Application to Natural Language Processing. *IBM Journal of Research and Development*, 32, pp. 170-184.

Bouma, Gosse (1990). Non-Monotonic Inheritance and Unification. W. Daelemans and G. Gazdar (Eds.) *Inheritance in Natural Language Processing Workshop Proceedings*, Institute for Language Technology and Artificial Intelligence, Tilburg University, Holland, pp. 1-8.

Brown, Charles, Veronica Dahl, D. Massam, Pierre Massicotte, and T. Pattabhiraman (1986). Tailoring Government and Binding Theory for Use in Natural Language Translation. Technical report LCCR TR 86-04, Simon Fraser University, Burnaby, BC, Canada.

Brown, Charles, T. Pattabhiraman, M. Boyer, D. Massam, and Veronica Dahl (1986). Tailoring Conceptual Graphs for Use in Natural Language Translation. N. Foo and J. Sowa (Eds.) *Conceptual Graphs in Knowledge-Based Systems*. Reading, MA: Addison-Wesley. Technical report LCCR TR 86-14, Simon Fraser University, Burnaby, BC, Canada.

Cercone, Nick J. (1980). The Representation and Use of Knowledge in an Associative Network for Automatic Comprehension of Natural Language. Leonard Bolc (Ed.) *Representation and Processing of Natural Language*. Munich, Germany: Carl Hanser Verlag, pp. 121-205.

Cercone, Nick J., Dan C. Fass, Gary Hall, Chris Groeneboer, Mimi Kao, Paul McFetridge, and Fred Popowich (1990). Using Relational Knowledge Structures to Handle Null Value Situations in Natural Language Interfaces. In *Proceedings of the 5th Rocky Mountain Conference on Artificial Intelligence (RMCAI-90)*, Las Cruces, NM, pp. 137-142.

Cercone, Nick J., Paul McFetridge, Gary Hall, and Chris Groeneboer (1989). An Unnatural Natural Language Interface. In *Proceedings of the Combined 16th International ALLC Conference & 9th International Conference on Computers and the Humanities*, Toronto, Ontario.

Cercone, N., R. Goebel, J. de Haan, and S. Miller (to appear). The ECO Family. In F. Lehmann (Ed.) *Semantic Networks in Artificial Intelligence*. Oxford, England: Pergamon Press. Also as a special edition of *Computers and Mathematics with Applications*, 23, (1-7), (Jan 1992).

Chomsky, Noam (1982). *Lectures on Government and Binding, The Pisa Lectures, 2nd (Revised) Edition* Dordrecht, Holland: Foris Publications.

Chomsky, Noam (1986). *Barriers*. Linguistic Inquiry Monograph 13, Cambridge, MA: MIT Press.

Crystal, David (1991). *A First Dictionary of Linguistics and Phonetics, 3rd Edition*. Boulder, CO: Westview Press.

Dahl, Veronica (1979a). Logical Design of Deductive, Natural Language Consultable Databases. *Proceedings of the 5th International Conference on Very Large Databases*, Rio de Janeiro, Brazil.

Dahl, Veronica (1979b). Quantification in a Three-Valued Logic for Natural Language Question-Answering Systems. *Proceedings of the 6th International Joint Conference on Artificial Intelligence (IJCAI-79)*, Tokyo, Japan.

Dahl, Veronica (1982). On Database Systems Development Through Logic. *ACM Transactions on Database Systems*, 7, (1), pp. 102-123.

Dahl, Veronica (1986). Logic Programming for Constructive Expert Systems. Larry Kerschberg (Ed.) *Expert Database Systems*. Benjamin/Cummings Series in Database Systems and Applications, Menlo Park, CA: Benjamin/Cummings.

Dahl, Veronica (1988a). Static Discontinuity Grammars for Government and Binding Theory. *Actes du colloque ILN'88 Informatique & Langue Naturelle*, LIANA, Université de Nantes, France. Also technical report CSS/LCCR TR 88-22, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Dahl, Veronica (1988b). Type Hierarchies for Deductive Databases. *Proceedings of the 8th Conference of the Brazilian Computing Society*. Rio de Janeiro, Brazil.

Dahl, Veronica (1989). Discontinuous Grammars. *Computational Intelligence*, 5, (4), pp. 161-179.

Dahl, Veronica (1990). Describing Linguistic Knowledge About Constraints in User-Friendly Ways. *International Journal of Expert Systems: Research and Applications*, 3, (2), pp. 131-146.

Dahl, Veronica (1991). Incomplete Types for Logic Databases. *Applied Mathematics Letters*, 4, (3), pp. 25-28.

Dahl, Veronica, Charles Brown, and Sharon Hamilton (1986). Static Discontinuity Grammars and Logic Programming. Technical report LCCR TR 86-17, Simon Fraser University, Burnaby, BC, Canada.

Dahl, Veronica, and Michael McCord (1983). Treating Coordination in Logic Grammars. *American Journal of Computational Linguistics*, 9, pp. 69-91.

Dahl, Veronica, and Fred Popowich (1990). Parsing and Generation with Static Discontinuity Grammars. *New Generation Computing*, 8, (3), pp. 245-274.

Dahl, Veronica, Greg Sidebottom, and Joerg Ueberla (1991). Automatic Design through Constraint-Based Reasoning. Extended abstract in *Proceedings of the International Logic Programming Symposium Workshop on Deductive Databases*, San Diego, CA, October 1991.

Evans, Roger, and Gerald Gazdar (Eds.) (1990). The DATR papers. Cognitive Science Research Paper CSRP 139, School of Cognitive and Computing Sciences, University of Sussex.

Flickinger, Dan, Carl Pollard, and Tom Wasow (1985). Structure-Sharing in Lexical Representation. *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, University of Chicago, Chicago, IL.

Fass, Dan C. (1988). Metonymy and Metaphor: What's the Difference? *Proceedings of the 12th International Conference on Computational Linguistics (COLING-88)*, Budapest, Hungary, pp. 177-181.

Fass, Dan C. (1989). Lexical Semantic Constraints. Technical Report CSS/LCCR TR 89-11, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada. To appear in J. Pustejovsky (Ed.) *Semantics and the Lexicon*. Dordrecht, Netherlands: Kluwer Academic Publishers.

Fass, Dan C. (1990). Four General Representations and Processes for Use in Problem Solving. In S. Ramani, R. Chandrasekar, and K. S. R. Anjeyulu (Eds.) *Knowledge Based Computer Systems (International Conference KBCS '89, Bombay, India, December 1989, Proceedings)*. Lecture Notes in Artificial Intelligence 444, Berlin, Germany: Springer-Verlag, pp. 169-178.

Fass, Dan C. (1991a). Met*: A Method for Discriminating Metonymy and Metaphor by Computer. *Computational Linguistics*, 17, (1), pp. 49-90.

Fass, Dan C. (1991b). Metonymy, Case Role Substitution and Sense Ambiguity. In Dan Fass, James Martin & Elizabeth Hinkelman (Eds.) *Proceedings of the IJCAI-91 Workshop on Computational Approaches to Non-Literal Language (Sydney, Australia)*, technical report CU-CS-550-91, University of Colorado at Boulder, Boulder, CO, pp. 42-51.

Fass, Dan C., Nick J. Cercone, Gary Hall, Chris Groeneboer, Paul McFetridge and Fred Popowich (1990). A Classification of User-System Interactions in Natural Language, with Special Reference to "Ill-Formed Input." In *Proceedings of the 5th Rocky Mountain Conference on Artificial Intelligence (RMCAI-90)*, Las Cruces, NM, pp. 143-148.

Fass, Dan C., and Gary Hall (1990). A Belief-Based View of Ill-Formed Input. In *Computational Intelligence '90*, Milan, Italy, September 24-28 1990. Also as technical report CSS/LCCR TR 90-18, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Fraser, Norman and Richard Hudson (1990). Word Grammar: An Inheritance-Based Theory of Language. W. Daelemans and G. Gazdar (Eds.) *Inheritance in Natural Language Processing Workshop Proceedings*. Institute for Language Technology and Artificial Intelligence, Tilburg University, Holland, pp. 58-64.

Gazdar, Gerald, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag (1985). *Generalized Phrase Structure Grammar*. Oxford, England: Basil Blackwell.

Groeneboer, Chris, and Nick J. Cercone (1991). A User Interface Architecture Based on a Constructivist Paradigm. Technical report CSS/LCCR TR 91-04, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Gupta, Ranabir, and Gary Hall (1991). Modeling Generation. *Proceedings of the 3rd International Workshop on Foundations of Models and Languages for Data and Objects*, Aigen, Austria.

Hadley, Robert F. (1990). Connectionism, Rule Following, and Symbolic Manipulation. *Proceedings of the 8th National Conference on Artificial Intelligence (AAAI-90)*, Boston, MA.

Hall, Gary, and Ranabir Gupta (1991). Modeling Transition. *Proceedings of the 7th International Conference on Data Engineering*, Kobe, Japan, Los Alamitos, CA: IEEE Computer Society Press, pp. 540-549.

Han, Jai-Wei (1991). Constraint-Based Reasoning in Deductive Databases. *Proceedings of the 7th International Conference on Data Engineering*, Kobe, Japan, Los Alamitos, CA: IEEE Computer Society Press, pp. 257-265.

Havens, W.S. (1990). Echidna Constraint Reasoning System: Programming Specifications. In *Computational Intelligence '90*, Milan, Italy, September 24-28 1990.

Havens, W.S., S. Sidebottom, G. Sidebottom, J. Jones, M. Cuperman, and R. Davison (1990a). Echidna Constraint Reasoning System: Next-Generation Expert System Technology. Technical report CSS-IS TR 90-09, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Havens, W.S., S. Sidebottom, M. Cuperman, R. Davison, S. Gaudet, and G. Sidebottom (1990b). Echidna Constraint Reasoning System: Programming Language Manual, Version 0. Technical report CSS-IS TR 90-07, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Joseph, Stefan, and Romas Aleliunas (1991). A Knowledge-Based Subsystem for a Natural Language Interface to a Database that Predicts and Explains Query Failures. *Proceedings of the 7th International Conference on Data Engineering*, Kobe, Japan, Los Alamitos, CA: IEEE Computer Society Press, pp. 80-87.

Kao, Mimi, Nick J. Cercone, and Wo-Shun Luk (1988). Providing Quality Responses with Natural Language Interfaces: The Null Value Problem. *IEEE* Transactions on Software Engineering, 14, (7), pp. 959-984.

Kaplan, Ronald, and Joan Bresnan (1982). Lexical-Functional Grammar: A Formal System for Grammatical Representation. Joan Bresnan (Ed.) *The Mental Representation of Grammatical Relations*. Cambridge, MA: MIT Press.

Karttunen, Lauri (1986). D-PATR: A Development Environment for Unification-Based Grammars. *Proceedings of the 11th International Conference on Computational Linguistics* and *Proceedings of*

the 24th Annual Meeting of the Association for Computational Linguistics, Institut fuer Kommunikationsforschung und Phonetik, Bonn University, Germany, pp. 74-80.

Lakoff, George, and Mark Johnson (1980). *Metaphors We Live By*. London, England: Chicago University Press.

McFetridge, Paul (1991). Processing English Database Queries with Head Driven Phrase Structure Grammar. Submission to the 2nd Japan-Australia Joint Symposium on Natural Language Processing.

McFetridge, Paul, and Nick J. Cercone (1990). The Evolution of a Natural Language Interface: Replacing a Parser. *Proceedings of Computational Intelligence '90*, Milan, Italy.

McFetridge, Paul, Nick J. Cercone, WoShun Luk, and Gary Hall (1988a). System X: A Portable Natural Language Interface. In *Proceedings of the 7th Biennial Conference of the Canadian Society for Computational Studies of Intelligence (CSCSI-7)*, Edmonton, Alberta, Canada, pp. 30-38.

McFetridge, Paul, Gary Hall, Nick J. Cercone, and WoShun Luk (1988b). Knowledge Acquisition in System X: A Natural Language Interface to Relational Databases. *Proceedings of International Computer Science Conference '88*, The Computer Society of the IEEE (Hong Kong Chapter), Hong Kong.

McTear, Michael (1987). *The Articulate Computer*. Oxford, England: Basil Blackwell.

Massicotte, Pierre (1988). Generating Conceptual Graphs from F-Structures. Master's thesis, Computing Science department, Simon Fraser University, Burnaby, BC, Canada.

Massicotte, Pierre, and Veronica Dahl (1988). Handling Concept-Type Hierarchies through Logic Programming. *Proceedings of the 3rd Annual Workshop on Conceptual Graphs*, held in conjunction with AAAI-88, St Paul, MN.

Matsumoto, Y. (1988). Parallel Parsing of Discontinuous Grammars. K. Fuchi and L. Kott (Eds.) *Programming of Future Generation Computers II*, Holland: Elsevier Science Publishers B.V. (North Holland).

Napier, H.A., D. Lane, R. Batsell, and N. Guadango (1989). Impact of a Restricted Natural Language Interface on Ease of Learning and Productivity. *Communications of the ACM*, 32, (10), pp. 1190-1198.

Nebel, Bernhard, and Gert Smolka (1989). Representing and Reasoning with Attributive Descriptions. IWBS Report No. 81, IBM Deutschland, Stuttgart, Germany.

Obermeier, Klaus K. (1989). *Natural Language Processing Technologies in Artificial Intelligence*. Chichester, West Sussex, England: Ellis Horwood.

Partee, Barbara H. (1972). Opacity, Coreference, and Pronouns. Donald Davidson and Gilbert Harman (Eds.) *Semantics of Natural Language*. Dordrecht, The Netherlands: D. Reidel Publishing Company, pp. 415-441.

Pattabhiraman, T., and Nick J. Cercone (1990a). Selection: Salience, Relevance and the Coupling between Domain-Level Tasks and Text Planning. *Proceedings of the Fifth International Workshop on Natural Language Generation*, Dawson, PA, pp. 79-86.

Pattabhiraman, T., and Nick J. Cercone (1990b). Evaluating Natural Language Generation Systems for Theoretical Merit. *Proceedings of the AAAI-90 Workshop on Evaluation of Natural Language Generation Systems*, Boston, MA.

Pattabhiraman, T., and Nick J. Cercone (1991). Salience in Natural Language Generation. *Proceedings of the IJCAI-91 Workshop on Decision Making throughout the Generation Process*, Sydney, Australia, 1991.

Pollard, Carl (1985) Lectures on HPSG. Unpublished lecture notes, CSLI, Stanford University, CA.

Pollard, Carl (1989). The Syntax-Semantics Interface in a Unification-Based Phrase Structure Grammar. S. Busemann and C. Hauenschild (Eds.) *Proceedings of the Workshop on GPSG and Semantics*, KIT-FAST Technical Report, Technical University, Berlin, Germany.

Pollard, Carl, and Ivan A. Sag (1987). *Information-Based Syntax and Semantics, Volume 1: Fundamentals*. Center for the Study of Language and Information, Stanford University, CA.

Pollard, Carl, and Ivan A. Sag (forthcoming). *Information-Based Syntax and Semantics, Volume 2: Topics in Binding and Control*. Center for the Study of Language and Information, Stanford University, CA.

Popowich, Fred (1989a). Tree Unification Grammar. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, Vancouver, BC.

Popowich, Fred (1989b). Reflexives in Head-Driven Phrase Structure Grammar. Technical report CSS/LCCR TR 89-13, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Popowich, Fred (1989c). A Tree Unification Grammar-Based Natural Language Processor. Technical report CSS-IS TR 89-08, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Popowich, Fred (1990). Tree Unification Grammar, Parsing and Inheritance Networks. Technical report CSS/LCCR TR 90-15 and CMPT TR 90-07, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Popowich, Fred, and Susan Kindersley (1991). Developing Lexical Unification-Based Grammars. *Actes du colloque ILN'91 Informatique & Langue Naturelle*, LIANA, Université de Nantes, France.

Popowich, Fred, and Carl Vogel (1990). Chart Parsing Head-Driven Phrase Structure Grammar. Technical report CSS-IS/CMPT TR 90-01, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Popowich, Fred, and Carl Vogel (1991). A Logic-Based Implementation of Head-Driven Phrase Structure Grammar. *Proceedings of the Third International Workshop on Natural Language Understanding and Logic Programming*, Swedish Institute of Computer Science, Kista, Sweden, pp. 239-255.

Proudian, Derek, and Carl Pollard (1985). Parsing Head-Driven Phrase Structure Grammar. *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, University of Chicago, Chicago, IL.

Reddy, Michael J. (1979). The Conduit Metaphor – A Case of Frame Conflict in Our Language about Language. Andrew Ortony (Ed.) *Metaphor and Thought*. London, England: Cambridge University Press, pp. 284-324.

Saint-Dizier, Patrick (1988). Contextual Discontinuous Grammars. Veronica Dahl and Patrick Saint-Dizier (Eds.) *Natural Language Understanding and Logic Programming II*, Holland: Elsevier Science Publishers B.V. (North Holland).

Saint-Dizier, Patrick (1990). Dislog: Programming in Logic with Discontinuities. *Computational Intelligence*, 6, (2). Also technical report LCCR TR 87-13, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Shieber, Stuart, Hans Uszkoreit, Fernando Pereira, Jane Robinson, and M. Tyson (1983). The Formalism and Implementation of PATR-II. B. Grosz and M. Stickel (Eds.) *Research on Interactive Acquisition and Use of Knowledge*. SRI International, Menlo Park, CA, pp. 39-79.

Sidebottom, Greg (1991). Satisfaction of Constraints on Integer Expressions. Grad Student Review, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Strzalkowski, Tomek, and Nick J. Cercone (1986). A Framework for Computing Extra-Sentential References. *Computational Intelligence*, 2, (4), pp. 159-180.

Strzalkowski, Tomek, and Nick J. Cercone (1989). Non-Singular Concepts in Natural Language Discourse. *Computational Linguistics*, 15, (3), pp. 171-186.

Sugimura, R., K. Hasida, K. Hatano, Y. Kubo, T. Okinushi, and T. Takizuka (1988). A Software Environment for Research into Discourse Understanding Systems. *Proceedings of the International Conference on Fifth Generation Computer Systems*.

Thomason, R. (1974). *Selected papers of Richard Montague*. New Haven, CT: Yale University Press.

Uszkoreit, Hans (1986). Categorial Unification Grammars. *Proceedings of the 11th International Conference on Computational Linguistics*, Bonn University, Bonn, Germany, pp. 187-194.

Vogel, Carl, and Fred Popowich (1990). Head-Driven Phrase Structure Grammar as an Inheritance Hierarchy. W. Daelemans and G. Gazdar (Eds) *Inheritance in Natural Language Processing Workshop Proceedings*, Institute for Language Technology and Artificial Intelligence, Tilburg University, Holland, pp. 104-113.

Vogel, Carl, Fred Popowich, and Nick J. Cercone (1990). Inheritance Reasoning for Head-Driven Phrase Structure Grammar. Technical report CSS/LCCR TR 91-07, Centre for Systems Science, Simon Fraser University, Burnaby, BC, Canada.

Centre for Systems Science

• Simon Fraser University established the **Centre for Systems Science** as a provincial centre for advanced study. The objective of CSS is twofold: coordinate and combine advanced systems research at SFU, and encourage application of research results within industry in order to further the technological development of British Columbia.

•CSS has assembled three research groups to streamline its efforts. The orientation of these groups is as follows: intelligent systems, microelectronics, and computer and communications systems. Because of shared research interests, this latter group merges with LCCR (**Laboratory for Computer and Communications Research**) for the publication of technical reports.

• CSS's three groups involve over 80 skilled faculty researchers principally from SFU's Schools of Engineering Science and Computing Science, but also draw substantially on Mathematics, Physics, Geography, Philosophy, Linguistics, Psychology, Communication, and Kinesiology.